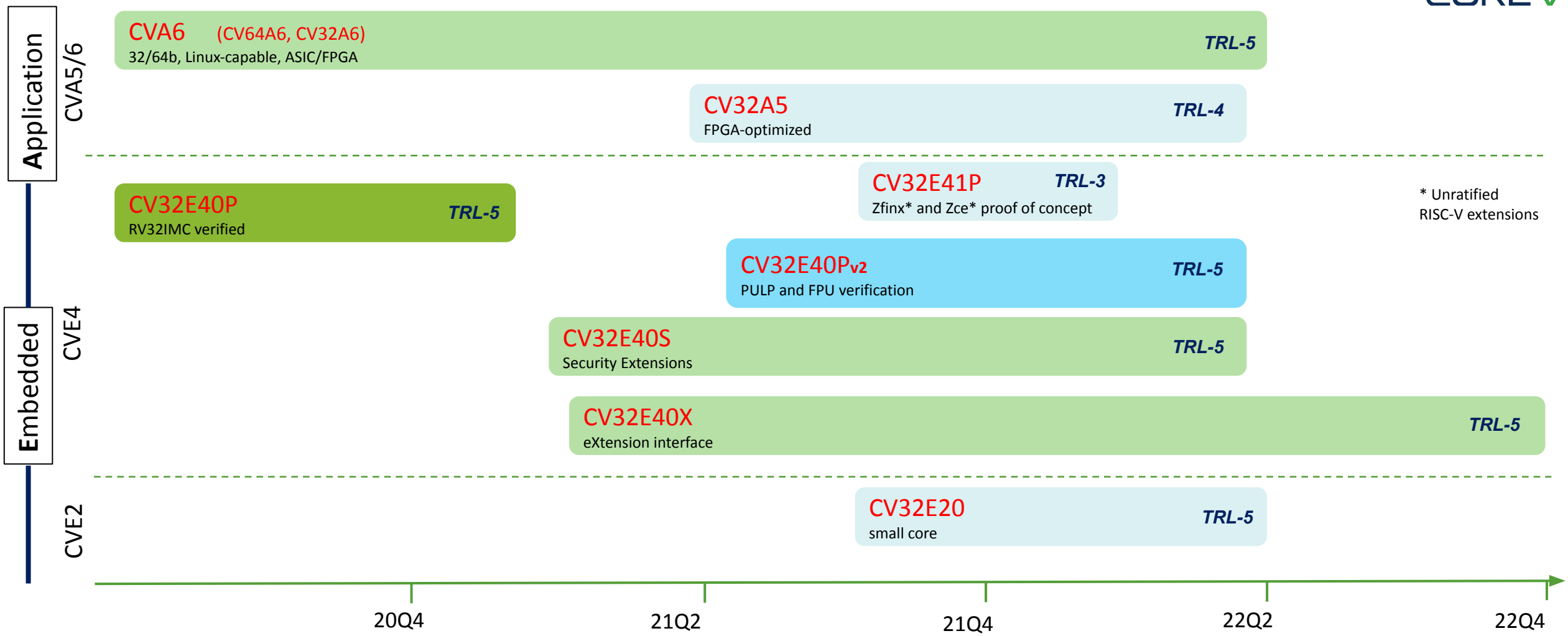
OpenHW Group CORE-V Cores Roadmap

Davide Schiavone, davide@openhwgroup.org
twitter: *@DavideSchiavo10*
Spring 2022 RISC-V Week

CORE-V™ Cores Roadmap



May 2022

CORE-V Cores

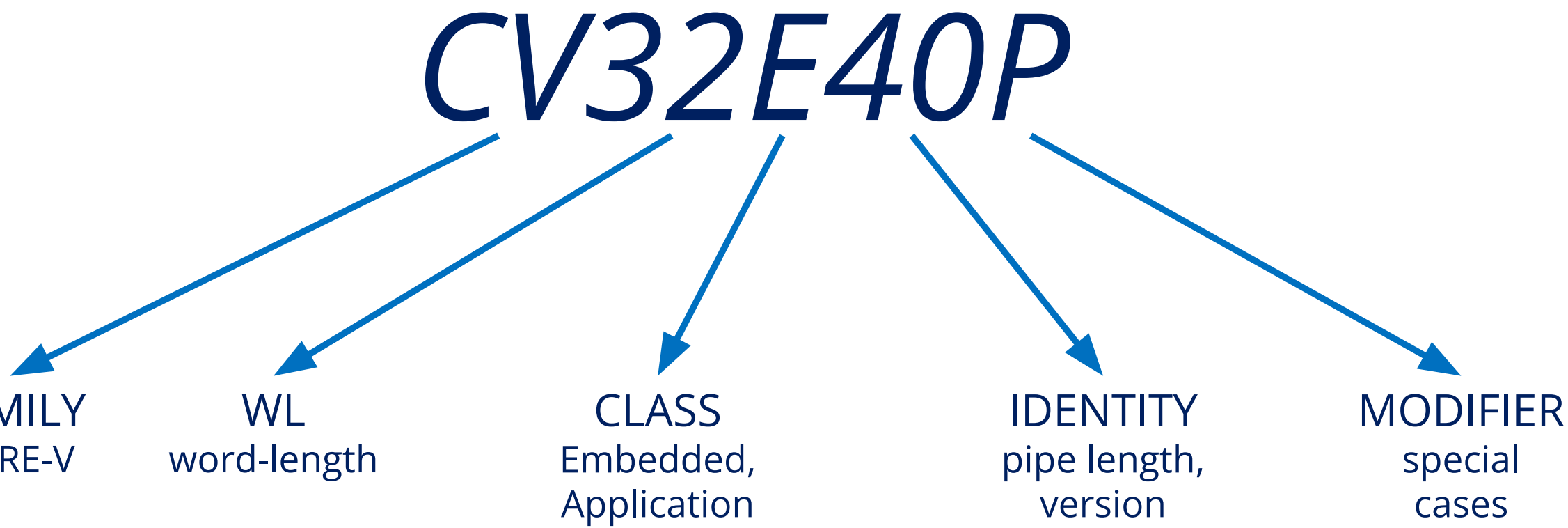
OpenHW Group is specialized in designing open-source, industrial-quality, competitive RISC-V cores

First two cores ***RI5CY*** and ***Ariane*** donated by ETH Zurich to take them from academic to industrial grades through

- Improved Documentation and repository structure
- Industrial Verification via the open-source ***core-v-verif*** project
- Maintenance rules to enable versions compatibility and support

The **CORE-V Family** continues to grow…

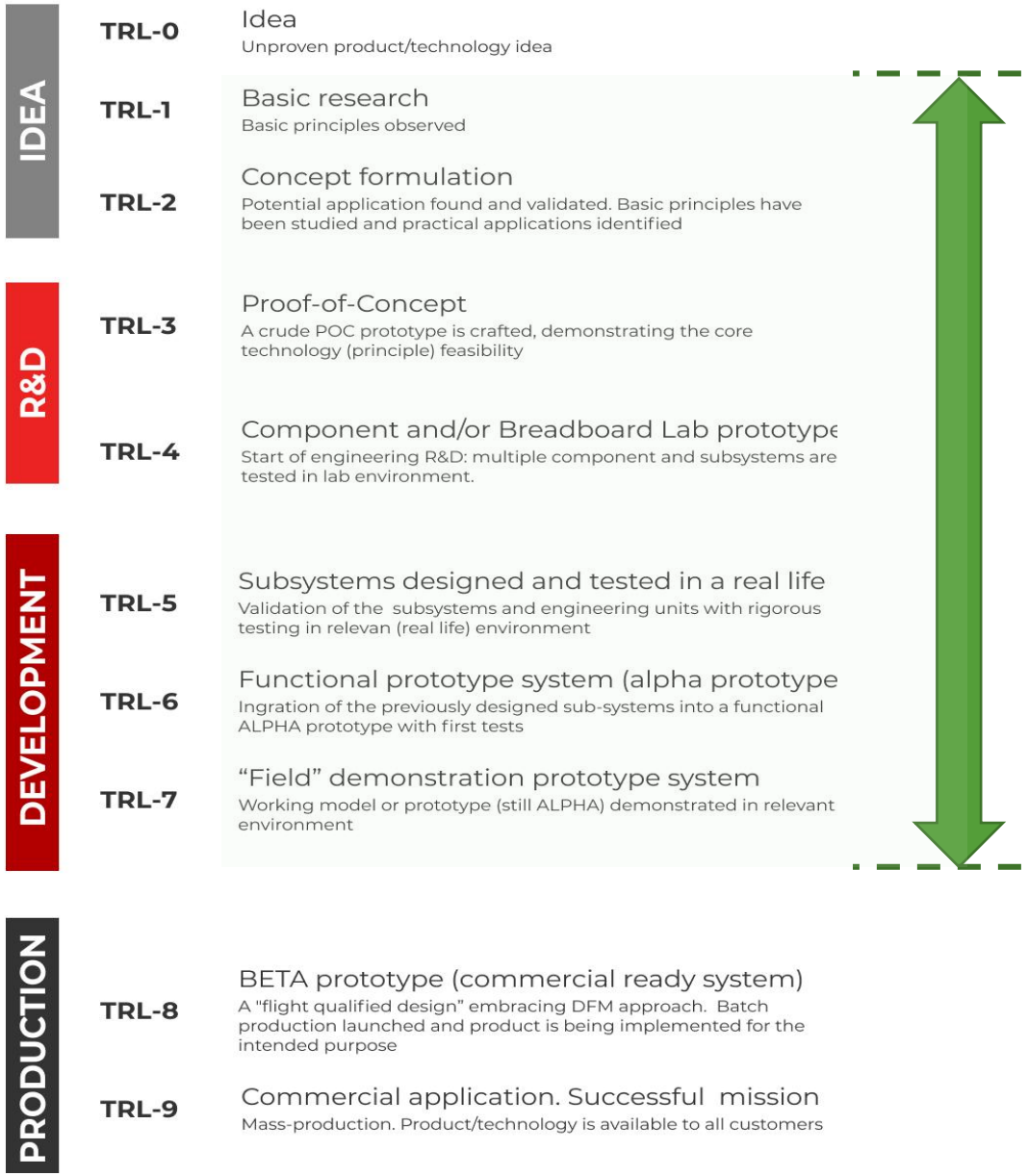
CORE-V Cores P/N Syntax



OpenHW Project Framework

Gate	Purpose	Criteria
PC Project Concept	Green-light of project concept by TWG	Proposed scope, initial view of the components and features, why do this project?
PL Project Launch	Full project launch approval by TWG	Outline of the requirements, features, components, project supporters, risks, high level schedule
PA Plan Approved	Communicate project plan to TWG, (allowing member participation and review	Project plan full checklist, project methodology. initial agile backlog, requirements specification
Project work		
PF Project Freeze	Completion of releasable project content	RTL Freeze checklist or other final checklist has been completed

TRL Scale



OpenHW Technology Outputs

OpenHW IP Adopters

CORE-V™

GROUP OPENHW™ PROVEN PROCESSOR IP

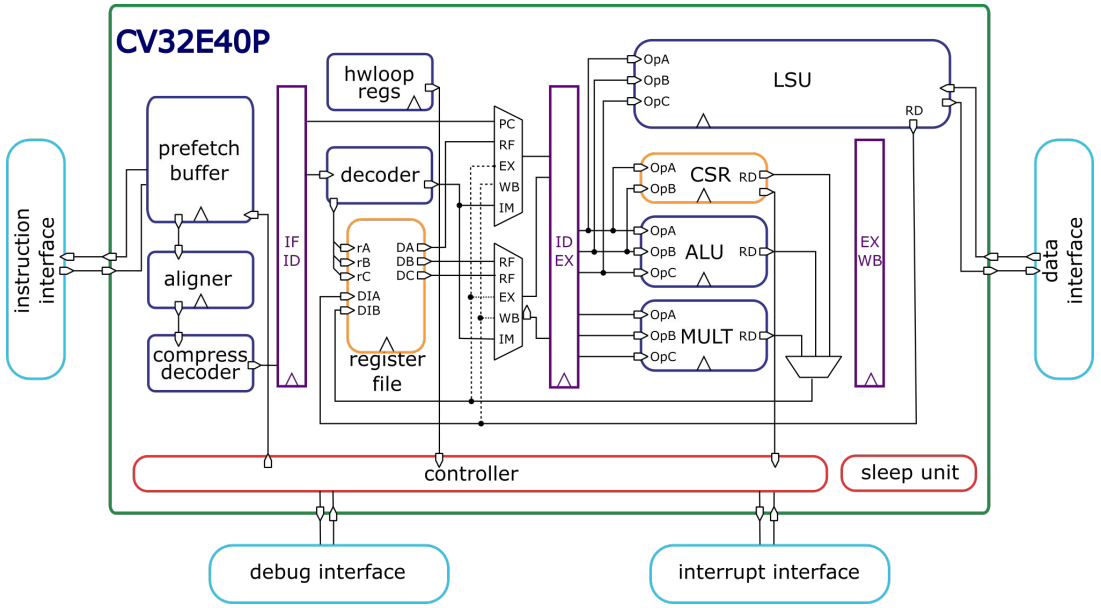
CORE-V Embedded-class Cores

CV32E40P – PF / TRL 5

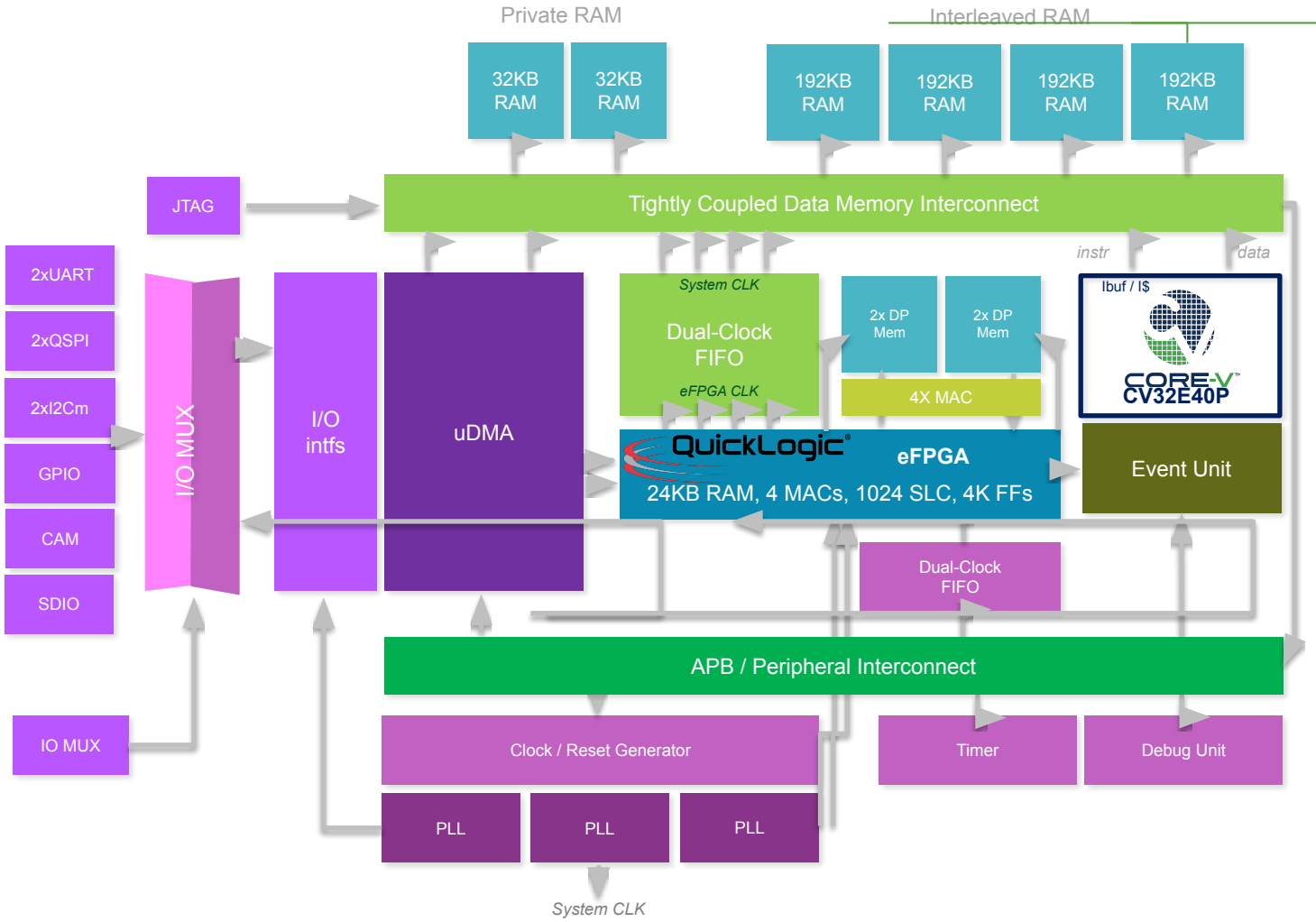
- 4-stage, in-order, single-issue
- RV32IM[F]CZicount_Zicsr_Zifencei [PULP_XPULP][PULP_CLUSTER][PULP _ZFINX]
- M-mode, CLINT, OBI

- ‘RTL Freeze’ achieved
  - RV32IMC extensions verified
  - Step&Compare with Imperas as reference model (100% coverage)
  - Interrupts and Debug

- Activities on RVFI interface
  - Facilitating sim-based step&compare verification FSM and formal verification



CORE-V™ MCU Tapeout 2nd half 2022



- Real Time Operating System (e.g. FreeRTOS) capable ~600+MHz CV32E4 MCU
- Embedded FPGA fabric with hardware accelerators from QuickLogic
- Multiple low power peripheral interfaces (SPI, GPIO, I2C, HyperRAM, CAMIF, etc) for interfacing with sensors, displays, and connectivity modules
- Built in 22FDX with GF

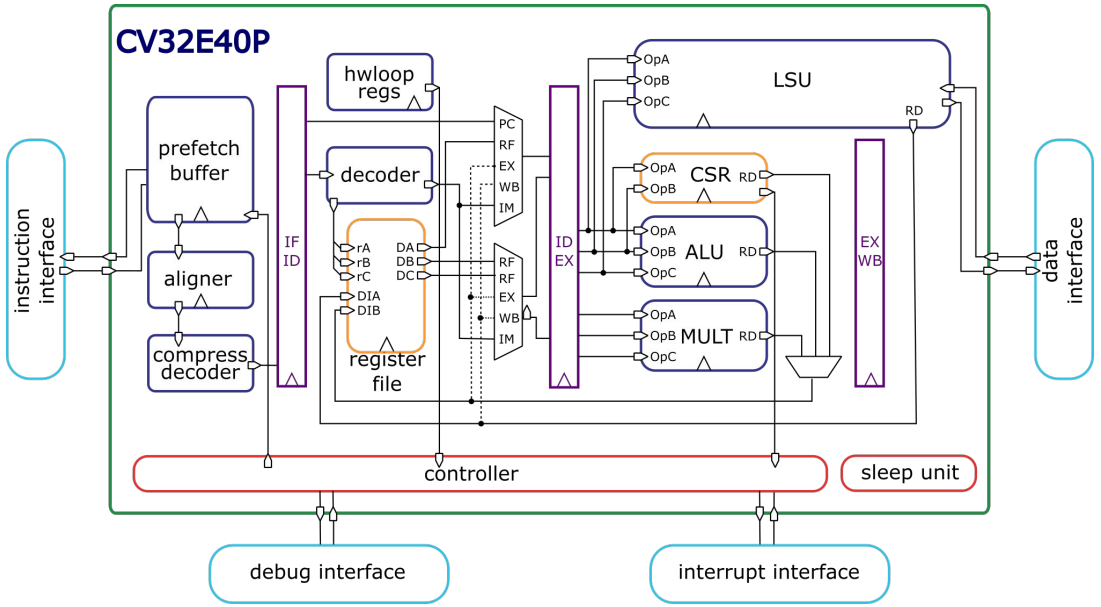
CV32E40P:v2 – PL / TRL 3

- RV32PULP_XPULP extensions
  - Verification and Reference Model
  - Moving the existing instructions to the RISC-V **custom space**
    - *use of custom-0, custom-1, custom-2, and custom-3*
  - SW support with upstream GCC and LLVM compiler

RV32F extensions

- Verification

- Project goal: industrial grade (TRL 5)



CI Flow to keep CV32E40P's sanity

As the RTL changes to the CV32E40P continue (e.g., moving to the custom instruction space, fixing bugs related to RV32F and RV32XPULP instructions), we need to ensure that the verified part of the core (RV32IMC) is kept **formally equivalent** (logical equivalent checking LEC)

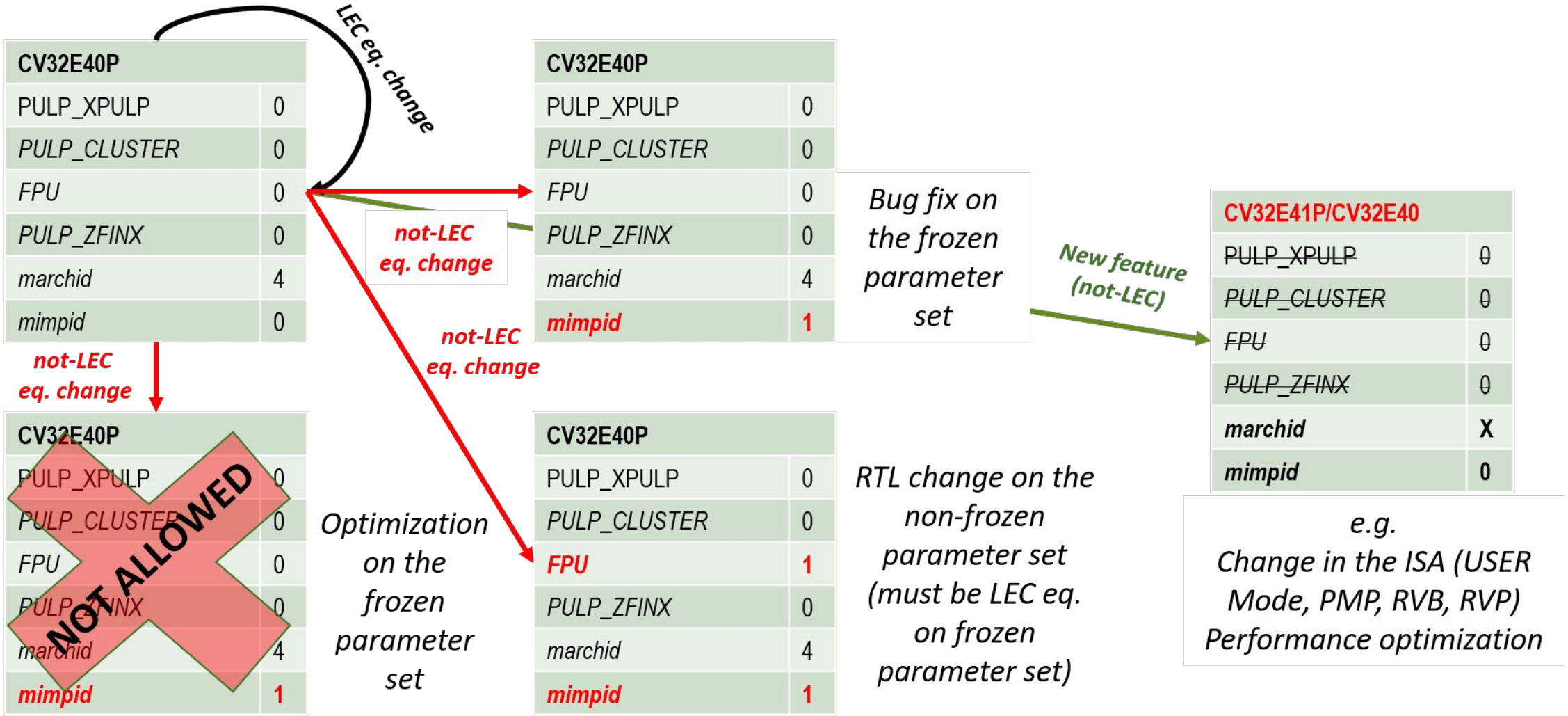
To achieve this, we DO NOT allow PPA optimizations or any other NON-LOGICAL-EQUIVALENT modifications to the verified part of the core

- this avoid compatibility issues and re-run of time-consuming verification routines

This is guaranteed by a **GitHub action running on AWS servers**

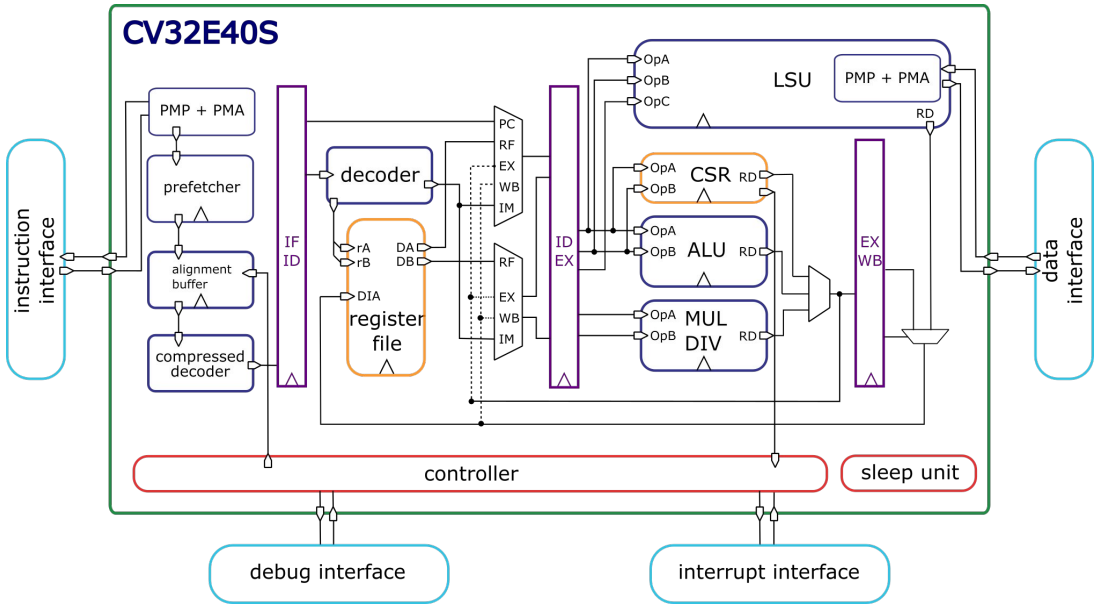
- each PR approved by OpenHW Group staff needs to pass the Logical Equivalent Checking script running on industrial EDA tools

Maintenance and Compatibility Rules



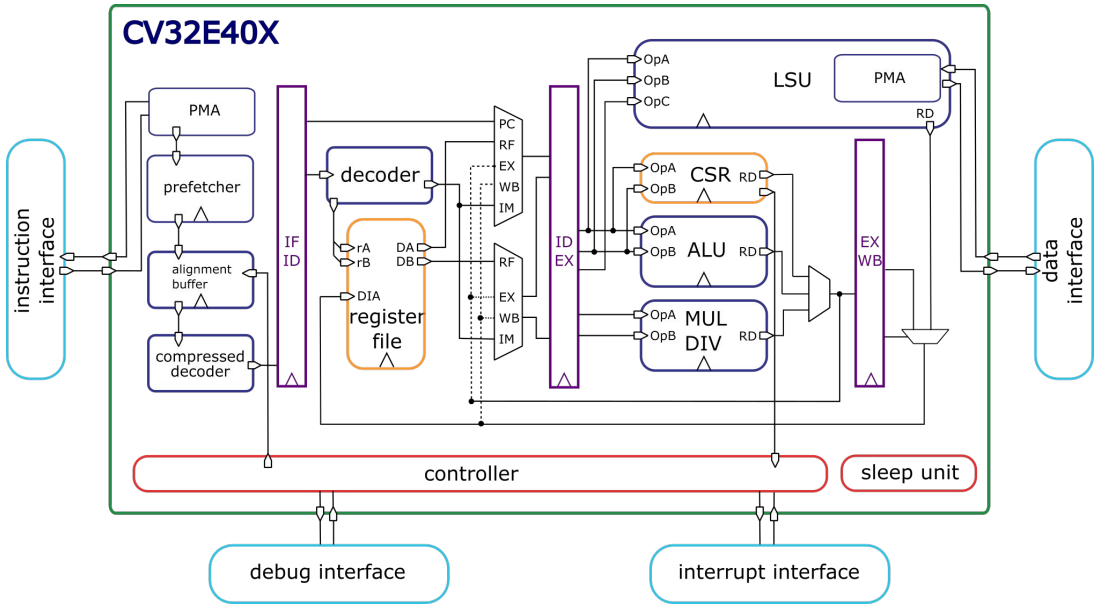
CV32E40S – PA / TRL 2

- 4-stage, in-order, single-issue
- RV32I[M|Zmmul]CXsecure[Zba_Zbb_Zbs|Zba_Zbb_Zbc_Zbs]Zceb_Zcee_Zces_Zicsr_Zifencei
- M/U-mode, CLINT, OBI, ePMP, PMA, bus error
- Secure core
  - Reduction of side-channel attacks
- Project goal: industrial grade (TRL 5)



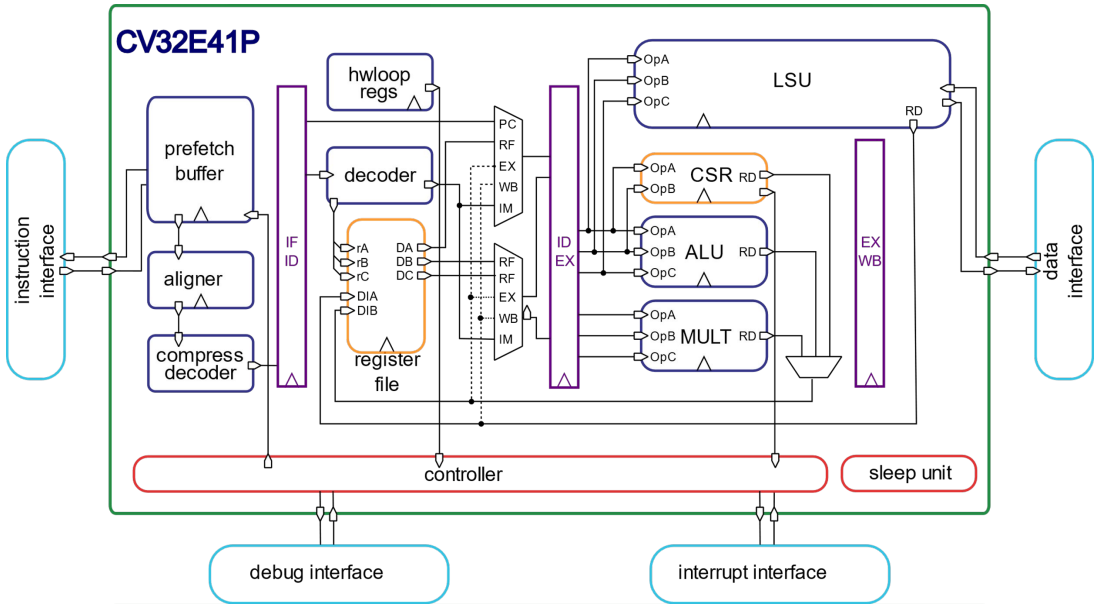
CV32E40X – PA / TRL 2

- 4-stage, in-order, single-issue
- RV32I[M|Zmmul][A]C[X][Zba_Zbb_Zbs|Zba_Zbb_Zbc_Zbs]Zceb_Zcee_Zces_Zicntr_Zicsr_Zifencei_Zihpm
- M-mode, CLINT, OBI, PMA, bus error
- Compute intensive core
  - CV-X-IF interface
- Project goal: industrial grade (TRL 5)



CV32E41P – PC / TRL 2

- 4-stage, in-order, single-issue
- RV32IMCZicount_Zicsr_Zifencei**[_Zce][{F, _Zfinx}]**[PULP_XPULP][PULP_CLUSTER]
- M-mode, CLINT, OBI
- Starting from CV32E40P fork
- Goals:
  - Proof of Concepts to demonstrate the PPA of Zce and Zfinx RISC-V draft ISA extensions
- Project goal: Proof of concept (TRL 3) *(next may be TRL 5)*

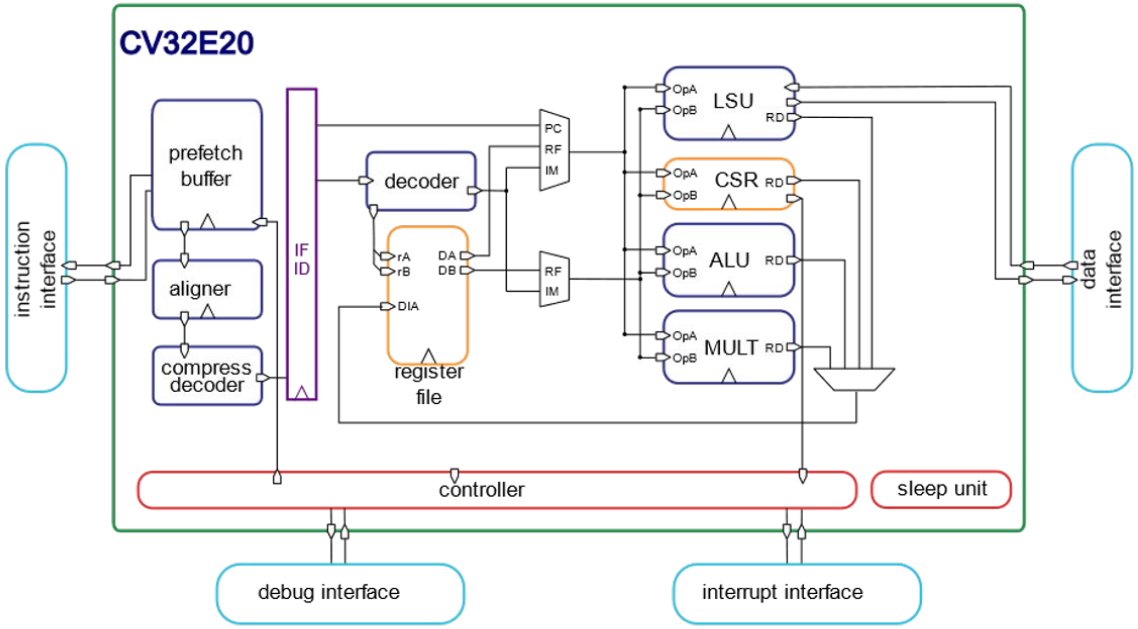


CV32E20 – PC / TRL 2

- 2-stage, in-order, single-issue
- RV32{I,E}[M]CZicount_Zicsr_Zifencei[_Zce]
- M-mode, CLINT, OBI

- Low area core
  - Optimized power and area for control-oriented applications
  - Starting point lowRISC Ibex (which started from ETH zero-riscy)
    - Clean-up parameters
    - Aligning IP interface with CV32E40* cores

- Project goal: industrial grade (TRL 5)



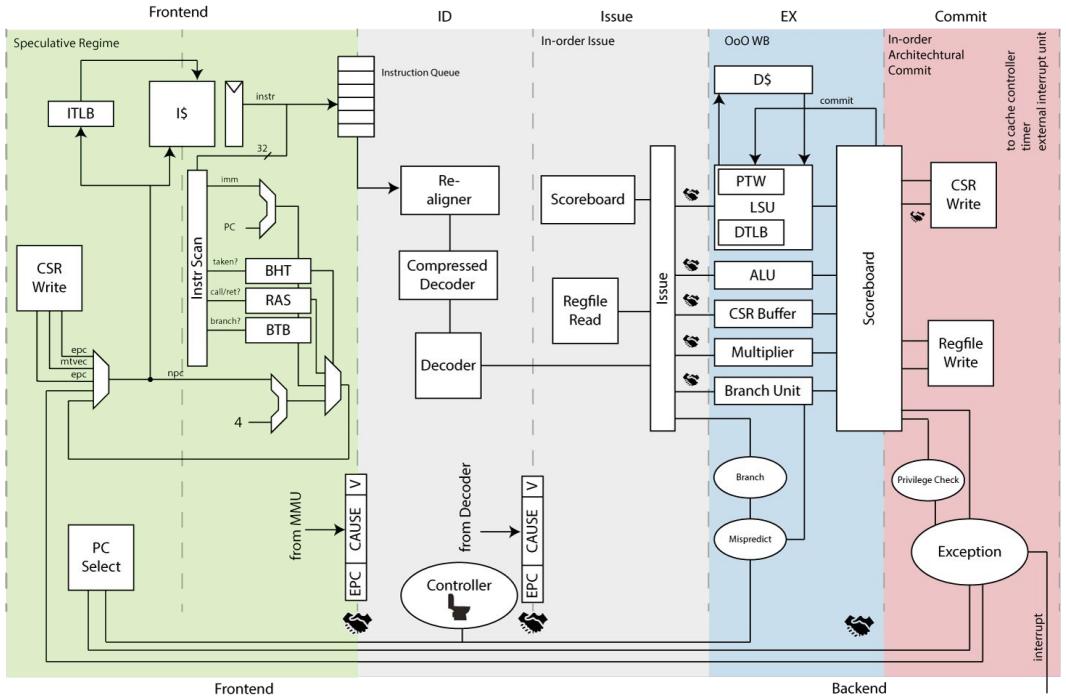
CORE-V Application-class Cores

CVA6

- 6-stage, in-order, single-issue
- RV{32|64}IMAC[FD]Zicsr
- M/S/U-mode, CLINT, AXI

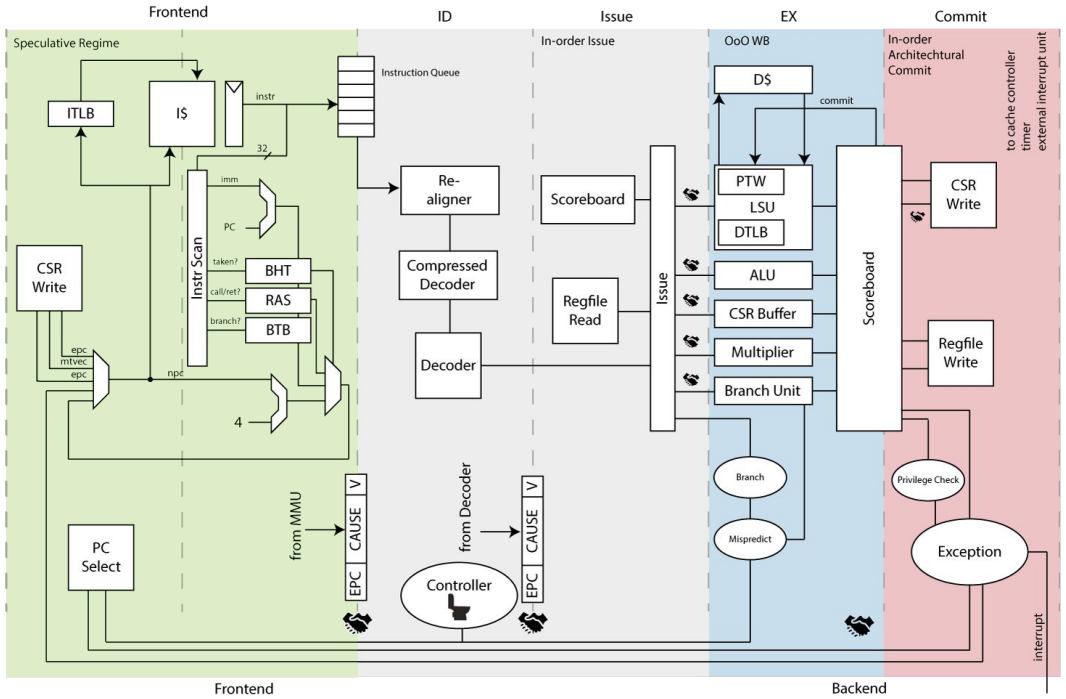
- Flexible application core
  - Linux-compatible thanks to MMU
  - 32 or 64 bit (CV32A6, CV64A6) from same RTL (64b from ETH, 32b from Thales)
  - L1 caches

- Project goal: industrial grade (TRL 5)
  - Currently drafting specifications, entry point for next stages



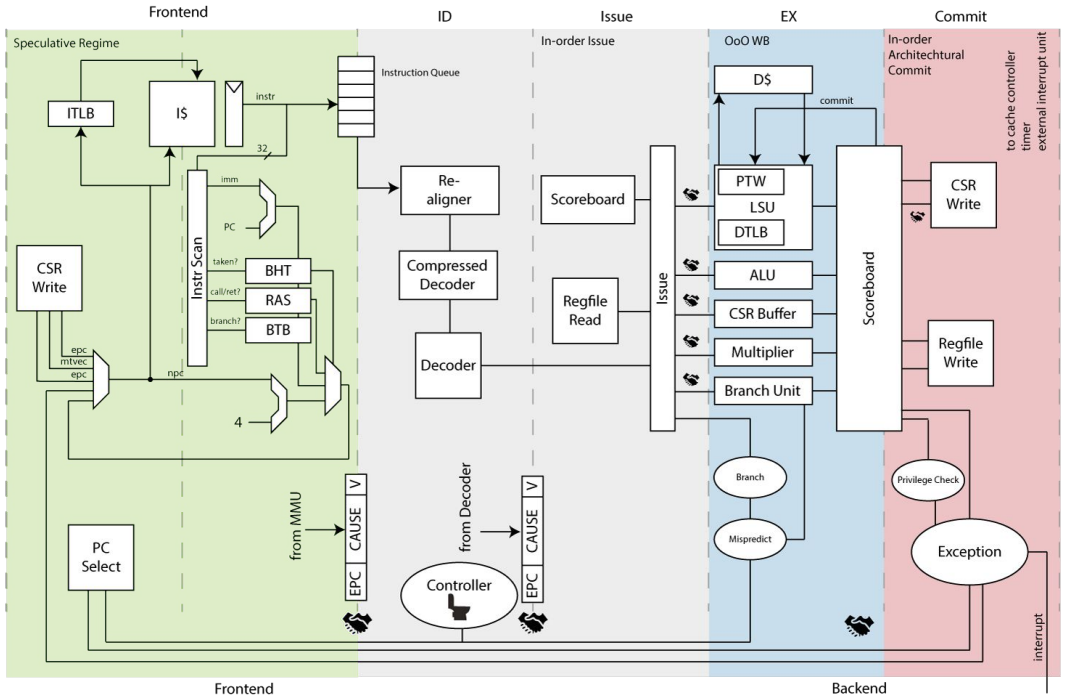
CV64A6 – PA / TRL 4

- Verification
  - RV64GC (RV64IMAFDC), debug, interrupts, privileges… to verify
  - sim-based Step&Compare, formal verification considered
  - RVFI interface
- New documentation
  - reStructured text-based as for the other cores
- FPGA optimizations
  - Target the Xilinx Genesys2, but not a Soft-Core!



CV32A6 – PA / TRL 3

- 32b version support for
  - Pipeline
  - MMU
  - Floating-point
  - Linux
- Verification
  - RV32IMAFC, debug, interrupts, privileges… to verify
  - sim-based Step&Compare, formal verification considered
  - RVFI interface
- PPA optimizations
  - ASIC
  - FPGA (vendor-independent soft-core)

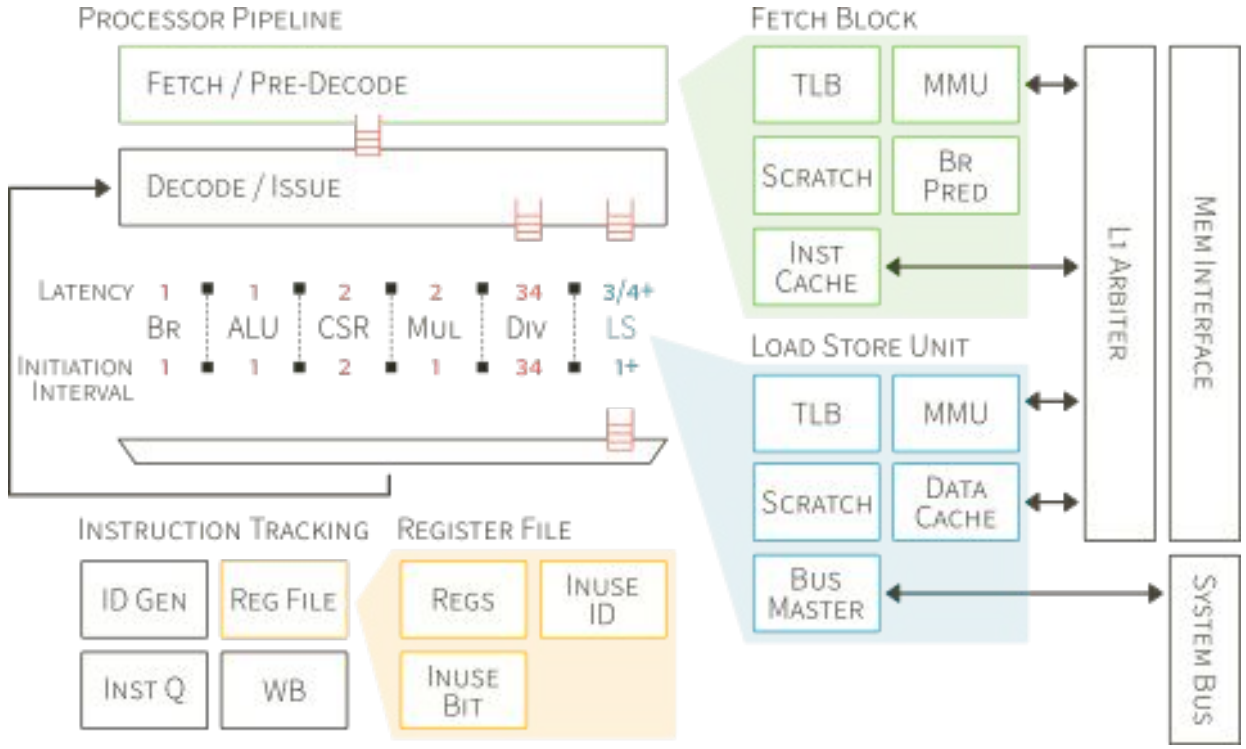


CV32A5 – PC / TRL 3

- 5-stage, single-issue
- RV32I[M][A]
- out-of-order executions
- M, [S,U] Privileged support

- Soft-core
  - Optimized for FPGA
  - Starting from TAIGA from Simon Fraser University

- Project goal: TRL 4

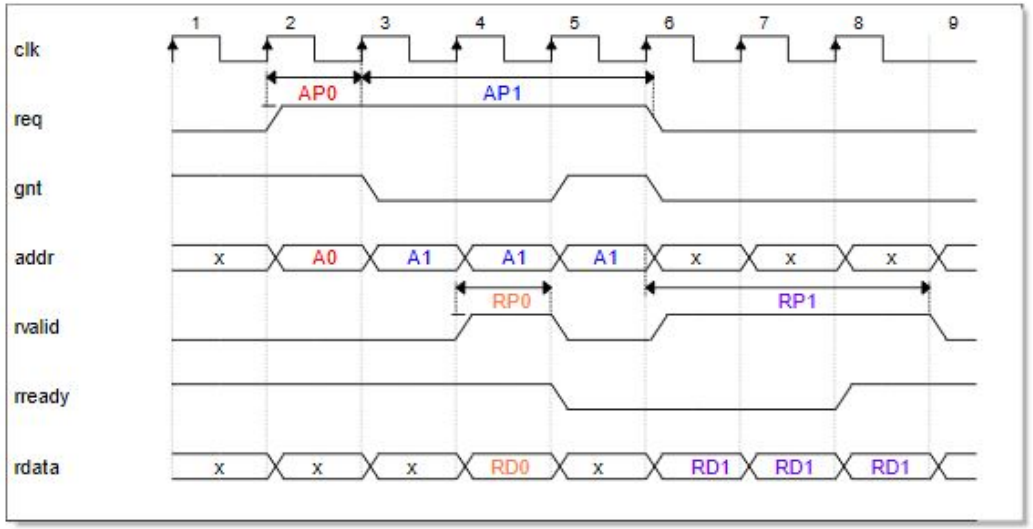


CORE-V SPECs

Specifications for RISC-V COREs

Open Bus Interface (OBI) Spec

- Memory Bus spec for RISC-V cores
  - Handshake based on ARM AMBA AXI
  - Coupled Read-Write channels
  - Easily translatable to ARM AMBA protocols
    - AXI, AHB
  - Based on: *https://raw.githubusercontent.com/openhwgroup/core-v-docs/master/cores/cv32e40p/OBI-v1.0.pdf*



CV-X-IF Spec

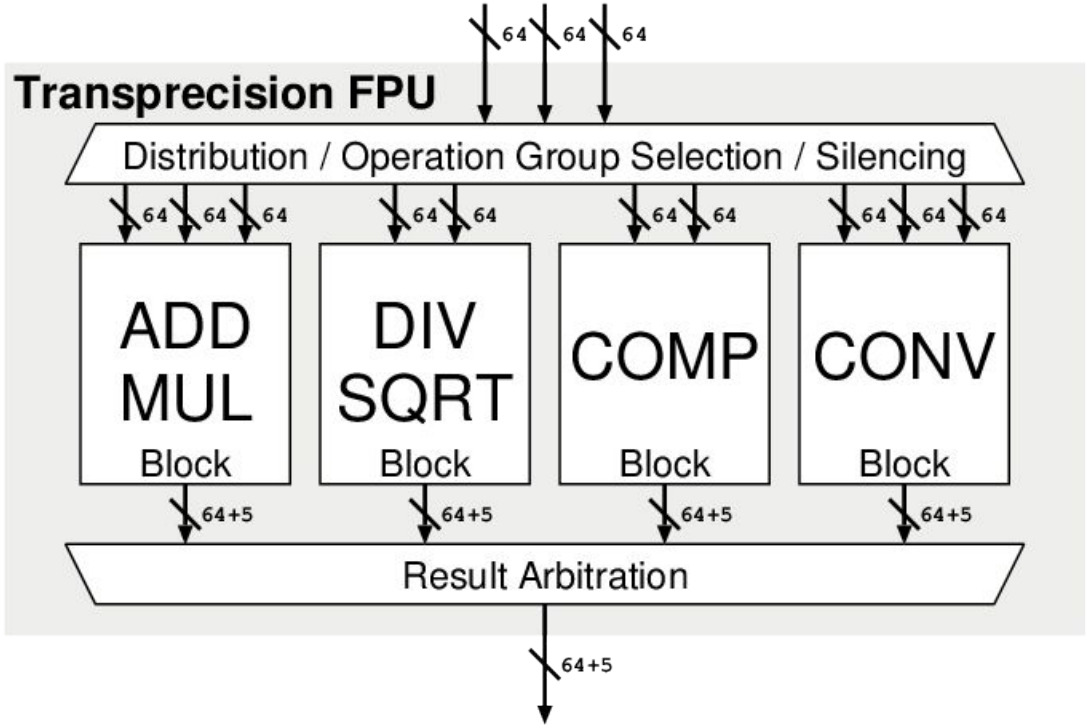
- Bus spec to allow RISC-V cores to offload RV extensions
  - CV32E40X will leverage this interface the most
  - CV32E40P and CVA6 will evaluate it
  - Based on: *https://github.com/openhwgroup/core-v-xif*
- Verification
  - Formal verification needed to test the spec
- Allows the cores to be agnostic about the custom extension definition

CORE-V Core Logic Blocks

IP surrounding the CORE-V COREs

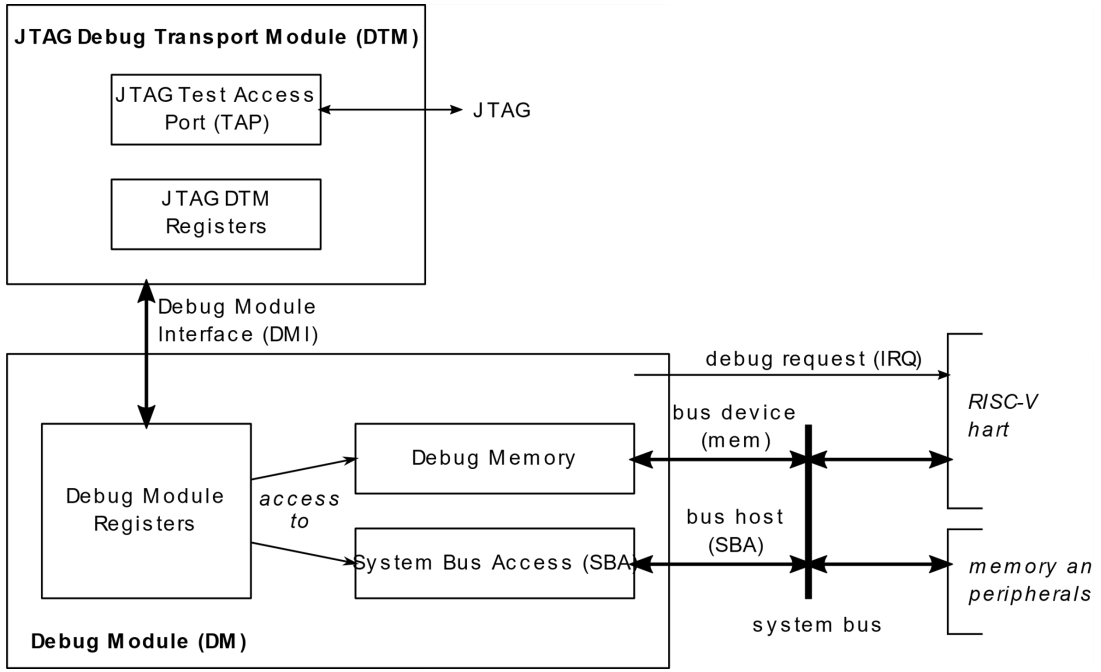
CV-FPU – PL / TRL 3

- Standalone co-processor that computes floating-point RISC-V RVF and RVD extensions
  - Parts of other projects as
    - CV32E40Pv2, CV32A6, CV64A6
- Possible starting point ETH *fpnew:*
  - Documentation
    - resTructured text documentation
  - Verification stand-alone or as part of the cores
- Project goal: industrial grade (TRL 5)



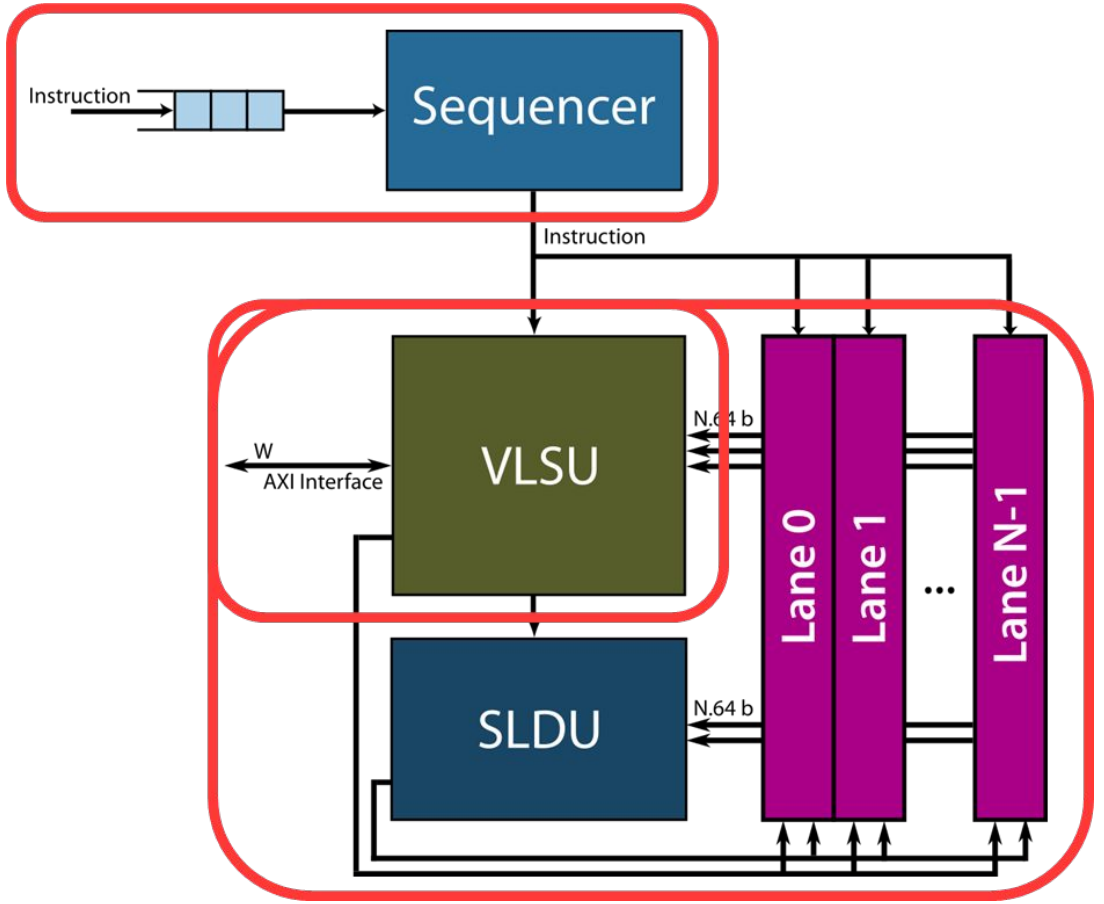
CV-DBG – PL / TRL 3

- Standalone IP that implements RISC-V Debug Module
  - Debug Transport Module, and ROM for Execution based debug
  - Parts of other projects as
    - CV32E40Pv2, CV32A6, CV64A6
- Possible starting point ETH *riscv-dbg:*
  - Documentation
    - resTructured text documentation
  - Verification stand-alone or as part of the cores
- Project goal: industrial grade (TRL 5)



CV-VEC – PC / TRL 3

- Datapath extensions that compute vectorial instructions based on RISC-V RVV extensions
- Starting point ETH ara
  - RVB vector extensions and mixed-precision
  - SW support for ML applications
  - Implementation in Globalfoundries 22FDX
  - Support for CV64A6
- Part of OpenHW Accelerate research program with Mitacs, CMC Microsystems, Polytechnique Montréal & ETH Zurich TRL 4



- OpenHW Group is a not-for-profit, global organization driven by its members and individual contributors where HW and SW designers collaborate in the development of open-source cores, related IP, tools and SW such as the CORE-V Family of open-source RISC-V cores

- OpenHW Group & CORE-V Family of open-source RISC-V cores for use in high volume production SoCs
  - Visit the OpenHW CORE-V Cores GitHub repository and contribute
  - Learn more at OpenHW TV

- Follow us on Twitter @openhwgroup & LinkedIn OpenHW Group